1. **REST, RESTful API**

+ REST là 1 kiến trúc phần mềm quy định các điều kiện và cách thức hoạt động của API.

+ RESTful API: Các api tuân thủ kiến trúc phần mềm REST

+ REST rule:

* Client-server: Mô hình client – server
* Stateless (phi trạng thái): Client và server không lưu trạng thái của nhau
* Cacheable (Khả năng caching): Các response có thể lấy ra từ cache 🡪 Server giảm tải xử lí request, client nhận được thông tin nhanh hơn
* Uniform interface (Chuẩn hóa interface): Hệ thống đặt ra các tiêu chuẩn để client dù là mobile, web đều có thể kết nối vào được
* Layered System (Phân lớp hệ thống): Chia tách các thành phần hệ thống theo từng lớp 🡪 Giảm độ phức tạp của hệ thống do các thành phần tách biệt nhau, từ đó dễ dàng mở rộng từng thành phần.

+ Example:

Một hệ thống RESTful API chứa tập các resources thông qua 1 tập các unique URI.

* https://example.com/api/v1/users
* https://example.com/api/v1/products

Trong ví dụ trên, users và products là các resources. Tên resource là danh từ số nhiều. https://example.com/api/v1/products : đúng

https://example.com/api/v1/getAllProducts : sai

1. **API RESTful hoạt động như thế nào?**
2. Client gửi request tới server
3. Server xác thực và xác nhận client có quyền thực hiện request không
4. Server tiếp nhận yêu cầu và xử lí
5. Server tạo response tới client
6. **Request API RESTful**

* Server xác định từng tài nguyên (resource) bằng mã định danh tài nguyên duy nhất. Đối với các dịch vụ REST, server thường thực hiện việc xác định tài nguyên bằng cách sử dụng Hệ thống định vị tài nguyên thống nhất (URL - Uniform Resource Locator). URL chỉ định đường dẫn đến tài nguyên. URL tương tự như địa chỉ trang web mà ta nhập vào trình duyệt. URL còn được gọi là request endpoint và chỉ định rõ ràng cho server những gì client yêu cầu.
* Method: Dev thường tạo các API RESTful bằng cách sử dụng giao thức HTTP.

Có 4 loại method HTTP phổ biến:

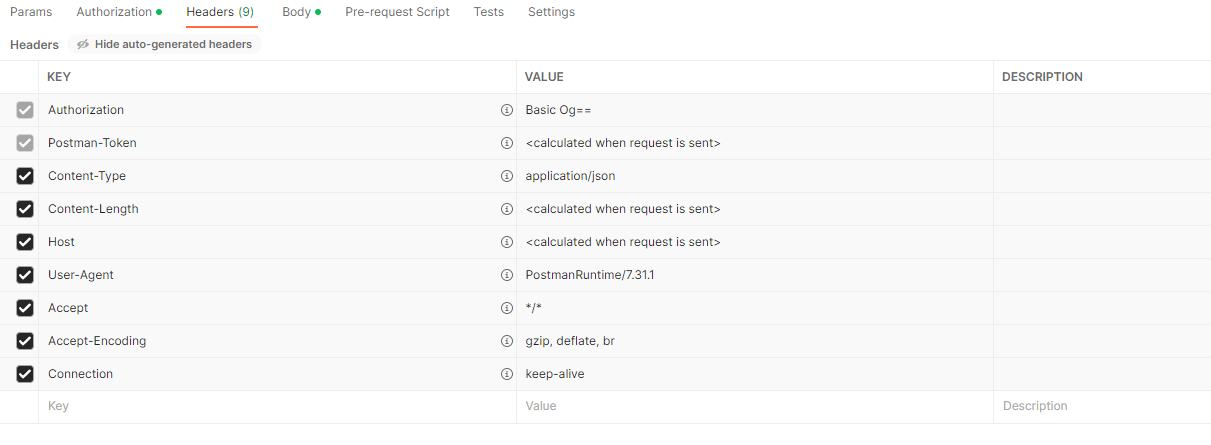
+ GET: Lấy tài nguyên có sẵn

+ POST: Tạo mới tài nguyên

+ PUT: Cập nhật tài nguyên

+ DELETE: Xóa tài nguyên

* HTTP Headers: Chỉ ra những thông tin trao đổi giữa client và server. Ví dụ như format của request và response.



+ Data: Data request client gửi lên

+ Params: Có thể gửi các request kèm theo params:

+ Query Params: Yêu cầu thêm thông tin về resource

+ Cookie Params: Xác thực client nhanh chóng

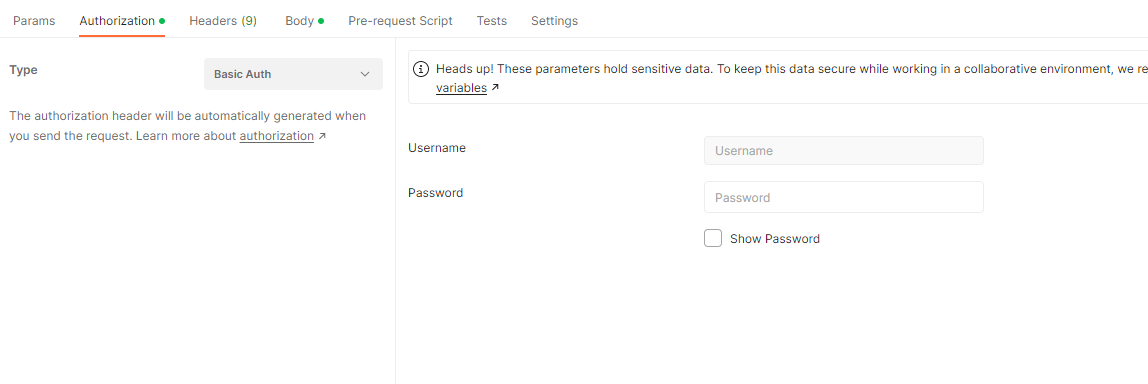
1. **Các phương thức xác thực API RESTful**

Dịch vụ web RESTful phải xác thực các yêu cầu trước khi gửi response về cho client.

Có 4 loại xác thực cơ bản:

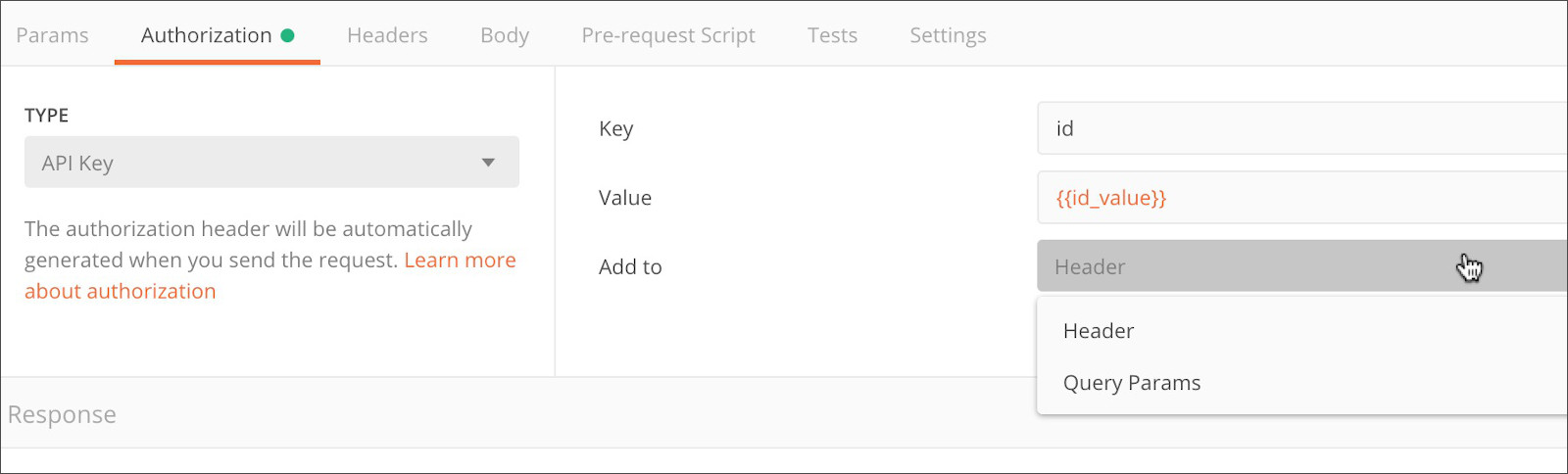
* HTTP authentication:

+ Basic authentication: client gửi username và password ở request header. Client mã hóa thông tin này bằng base64.



+ Bearer authentication: Server tạo ra 1 chuỗi token sau yêu cầu đăng nhập. Client thêm token này trong Authorization header.

* API keys: Một cặp key-value được bao gồm trong header hoặc query parameter sẽ gửi tới server khi gọi API. Dùng cách này không bảo mật.



Query params: GET /endpoint?api\_key=api\_value

Header: Api-Key: api\_value

* OAuth: (nghiên cứu sau)

1. **Response API RESTful**

REST yêu cầu server trả về những thành phần chính sau:

* Status:

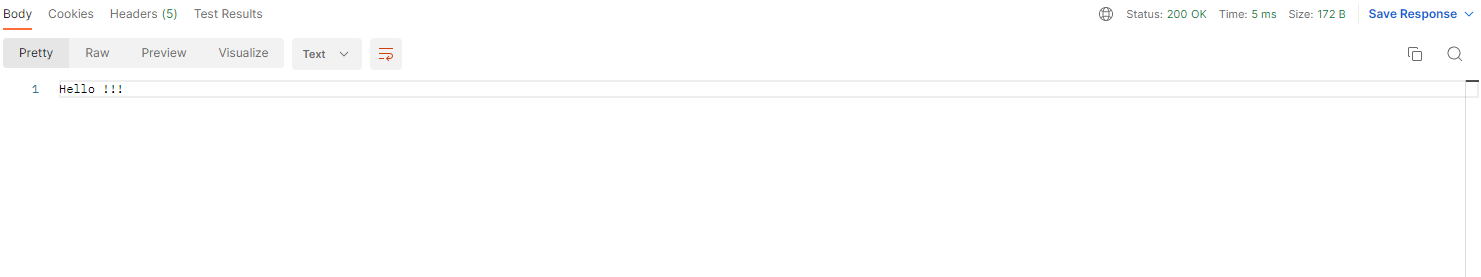
+ 200: Thành công

+ 400: Incorrect request

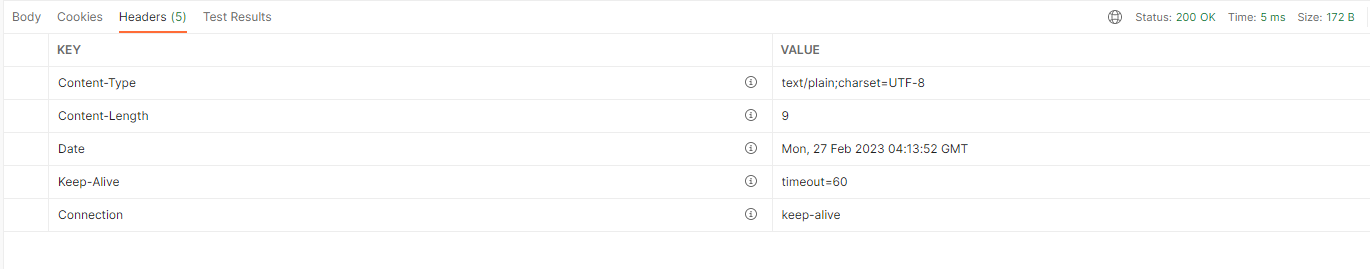
+ 404: Resource not found

+ 500: Internal Server Error

* Message Body:



* Header:

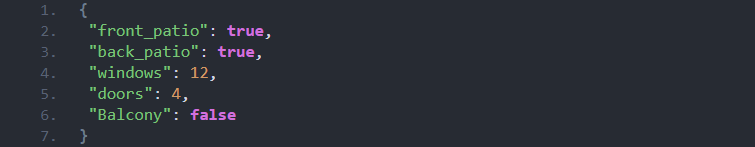


1. Câu hỏi:
   * + 1. PUT/PATCH khác gì nhau?

+ PUT: Dùng để update toàn bộ resource.

+ PATCH: Dùng để update 1 phần resource.

Ví dụ: Ta có 1 đối tượng



Dùng PUT/PATCH:



Kết quả:

+ Dùng PUT: 

+ Dùng PATCH:



Khi 1 resource không tồn tại, nếu dùng method

+ PUT: resource mới được tạo ra, lúc này tương đương với dùng POST..

+ PATCH: Fail, không có resource nào được tạo ra.